GUARANTEED ACCURACY COMPUTATIONS IN SYSTEMS AND CONTROL

M. Kanno and M. C. Smith

Department of Engineering, University of Cambridge
Trumpington Street, Cambridge, CB2 1PZ, United Kingdom

Keywords: Computer algebra, interval methods, numerical methods, norm computations, controller synthesis.

Abstract

This paper is concerned with the problem of validation in the context of numerical computations in control. We explore the possibility of using computer algebra tools and interval methods to compute solutions which have guarantees on accuracy, e.g. which are not subject to unknown errors due to rounding or approximation. We demonstrate that this is possible for two common norms of a linear system ($\mathcal{L}_2$ and $\mathcal{L}_\infty$) and $\mathcal{H}_2$-optimal controller synthesis. We further discuss some of the issues involved in achieving a validation property for other problems of controller synthesis.

1 Introduction

The aim of this work is to investigate the development of numerical methods for systems and control which have a guarantee on accuracy. An end-product of such work is an algorithm which could be described as 'infallible' in the following sense: the user would specify *a priori* a tolerance as small as desired, and the computer would provide an answer which was guaranteed to be accurate to the specified tolerance. A characteristic feature of such work is the application of computer algebra tools and the avoidance of floating-point arithmetic. This direction is an established subject within computer science, as well as a few application areas in science and engineering (see [1]), however it has not received attention so far in the control systems area.

The current *status quo* for numerical computation in control, which is based heavily on standard linear algebra routines and high-speed floating point arithmetic, is unable to satisfy the rigorous requirement on accuracy indicated above. At first sight, one might argue that a plant transfer function or state-space realisation is never known with complete accuracy, and therefore small (rounding) errors arising during calculation do not matter, and occasional larger errors are occupational hazards which rarely cause major problems. We claim that this is an unsatisfactory view for the following reasons.

1. In the majority of cases the user has no way to determine the extent to which errors have accumulated during calculation. In other words, moderate or large errors might go unnoticed.
2. Standard approaches to error analysis in numerical analysis do not solve the problem, since error estimates (rarely used in practice anyway) are themselves subject to rounding errors and can be erroneously computed [17].
3. The role played by computer power in this issue is deceptive. To repeat a calculation at successively higher precisions does not give a guarantee on accuracy, even when the solutions appear to converge [17]. In practice, increased computer power is often deployed instead for more involved calculations. Thus, the problem is not likely to decrease as computer power increases (and might even increase), if the current calculation methods persist.
4. Lack of attention to accuracy of solution has quite possibly retarded the development of effective software tools in certain areas. The example of ℓ_1-control is a case in point.
5. Although a 'good engineer' will always make checks whenever possible on final answers by contrasting methods to verify that they are 'sensible', there is an advantage if one source of errors can be eliminated. Moreover, it is possible that computational instabilities can prevent the desired answer from being found at all.
6. Although a solution to an engineering problem which is highly sensitive to the data is not an acceptable engineering solution, it is important not to confuse this idea with the numerical sensitivity of a particular algorithm. It is perfectly possible for a solution (e.g. a controller that one computes) to have low sensitivity to the given data (e.g. plant parameters) and to be a good engineering solution, but for the algorithm one uses to compute the solution to be highly sensitive to the data. With such an algorithm there is a clear difficulty in finding the solution one wants.

The above reasons are typical of the arguments that have been accepted in computer science to justify research in 'validated numerical algorithms'. The authors believe that these arguments also hold good in the context of computational methods for control systems.

2 A Couple of Software 'Glitches' for Illustration

Let us take the following numerical example system and try to find its $\mathcal{H}_\infty$-norm:

$$G(s) \quad = \quad \frac{s^2 + 10^{-7}s + 1}{s^2 + 10^{-8}s + 1}\,. \tag{1}$$

By hand calculation, it is easy to show that $\|G(s)\|_\infty = 10$ (with the norm being achieved at $s = j$). However, *hinfnorm*, a Mat-

lab command to compute the $\mathcal{H}_\infty$-norm based on a quadratic convergent method proposed in [3, 4], fails to converge and thus fails to find the norm:

```
>> out = hinfnorm(nd2sys([1 10^-7 1],
                         [1 10^-8 1]))
HINFNORM iteration DID NOT converge
a lower bound for the norm is 10
out = 10.00000001492336  Inf  1.00000000000000
```

An old-fashioned slow converging bisection method implemented by using existing Matlab functions converges and finds the $\mathcal{H}_\infty$-norm between 10.2041 and 10.2042 (the lower bounds is in fact above the actual $\mathcal{H}_\infty$-norm):

```
>> out = hinfnorm_bs(nd2sys([1 10^-7 1],
                           [1 10^-8 1]))
out = 10.20414352416992  10.20420074462891
```

Let us take another example. Consider the synthesis problem of finding the gap-optimal controller for $P = \frac{s-0.02}{s^2+1}$. Using the *ncfsyn* command in Matlab:

```
>> P = nd2sys([1 -0.02], [1 0 1]);
>> [Kopt, emax] = ncfsyn(P, 1)
Kopt =   1.02019794021444
emax =   0.70000214192573
>> [er1, er2] = emargin(P, Kopt)
SYSTEM has closed-right-half plane poles
er1 =     0
er2 =   Inf   Inf     0
```

The command returns a controller without any warnings, so the user would believe that there were no difficulties in the computation and that the controller is reasonably close to the actual one. Nevertheless the controller is in fact NOT stabilising.

3 Previous Work

As pointed out above the idea of 'validated numerical methods' or 'guaranteed accuracy' is not new in the computer science field. In [17], Krandick and Rump elucidate this idea as a search for algorithms with a rigorous specification, i.e., methods that never fail. Several papers in [16] propose hybrid symbolic-numerical approaches to validation. The underlying idea is 'to begin with infallible algorithms and to make them faster' [5]. Conventional researches on numerical methods are 'to make fast algorithms less fallible' [5]. We employ the former idea to solve problems in systems and control with a guarantee.

Two useful methodologies for validated numerical methods which are exploited in this work are interval methods and polynomial root localisation. Interval methods have been used to examine the existence of a solution to a system of equations in an interval and further to find a sequence of intervals which converge to a solution [18, 22] (a detailed treatment of some background theory on interval methods and applications can be found in [13, 21, 26]). Using rationals as the underlying number system these methods have the potential to localise solutions with a guarantee. Although typical polynomial root solvers employ floating point arithmetic, there are some standard techniques which can be used to find roots with guaranteed accuracy. These will be discussed further in Section 6.

The increasing interest in the use of symbolic methods in control systems is illustrated in [23, 24] and one of a variety of different objectives in those works is to improve numerics. Some techniques such as Quantifier Elimination, Groebner bases and Bernstein expansion have been applied to control problems, which are illustrated in [11, 12, 27, 31]. It may be interesting to point out that many of the articles in [12] are in fact related to control problems. However, so far, there has been no attempt to devise validated numerical methods for control in the precise sense of Krandick and Rump.

4 Number Systems

The manner of representing real numbers on a computer goes to the heart of the issue of accuracy in numerical calculations. In *floating point* arithmetic there is a mantissa and exponent with fixed length. The advantage is that arithmetical operations can be computed in fast hardware. Nevertheless the limited accuracy, i.e., sparseness, prevents one from finding a solution with arbitrarily accuracy. More serious problems are that the floating point system is not closed under arithmetical operations and that the operations are neither associative nor distributive.

This problem is recognised in computer algebra packages and the rational number system is provided to do exact arithmetical operations. Even though the rational number system is not complete, rational numbers satisfy the axioms for algebraic operations and for an ordered field. Moreover, since rational numbers are dense in the reals, they are suitable for finding solutions with arbitrary accuracy. These advantages are achieved at the price of slower arithmetical operations and the need for 'dynamic' data structures to store numbers.

5 The Meaning of Guaranteed Accuracy

When solving a problem which finds a single real number (e.g. $\mathcal{H}_\infty$-norm, stability margin etc), a guaranteed accuracy algorithm has to use a computer representable number system and also produce an interval, which is a pair of elements in the number system used, to bound the true answer. The following formal definition is thus suggested.

Definition 1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be well-defined (not necessarily continuous). Let A be some given algorithm taking the form of an executable procedure, which generates a well-defined function $A : (\mathbb{F}^n, \mathbb{F}) \to \mathbb{F}^2$ where $\mathbb{F} \subset \mathbb{R}$ is a set of computer representable numbers and $A(\mathbf{P}, \varepsilon) = (f_\ell, f_r)$ where $f_\ell < f_r$. Then, A is said to be a guaranteed accuracy algorithm for f over $\mathbb{F}$ if, for any $\mathbf{P} \in \mathbb{F}^n$ and any $\varepsilon \in \mathbb{F}$, $\varepsilon > 0$, the true $f(\mathbf{P})$ is contained in the closed interval $[f_\ell, f_r]$ and $f_r - f_\ell < \varepsilon$.*

In our approach, the rational number system is used, i.e., $\mathbb{F}$ in the definition will always be taken to be $\mathbb{Q}$.

The above definition is reminiscent of a standard approach to

computing the $\mathcal{L}_\infty$-norm by a bisection method. However there is an important difference to standard implementations using floating point arithmetic in that it is mandatory that the statement $f(\mathbf{P}) \in [f_\ell, f_r]$ refers to the 'true' real number $f(\mathbf{P})$, and not some approximation of it. We point out that the intention of the Matlab algorithms in Section 2 is to produce an interval containing the true $\mathcal{H}_\infty$-norm, but there is a failure to do so because of rounding errors.

6 Polynomial Root Localisation

Polynomial root solvers typically employ floating point arithmetic, and as such, do not find roots with guaranteed accuracy. For a real polynomial there is a well-known approach to localise the *real* roots using the method of Sturm chains [8] which is very suitable for a guaranteed accuracy implementation. In computer algebra packages this is often implemented using Descartes' rule of signs suggested in [2]. Several real root localisation algorithms using the above two methods and others are discussed and calculation speed comparisons are made in [6].

It is also possible to do root localisation for the complex roots of a polynomial with guaranteed accuracy via the Lehmer-Schur method [19], which examines the existence of roots of a polynomial inside the unit circle. By a suitable transformation one can examine whether a root is inside a circle centred at an arbitrary location and with an arbitrary radius. Some methods of determining whether a rectangle in the complex plane contains a root are suggested in [5, 7, 28, 30].

Guaranteed root localisation methods will find an application in the present work for various substeps in the algorithms to be developed. One example will be the problem of factoring a polynomial into a product of a stable factor and an anti-stable factor with guaranteed accuracy. In particular for a monic polynomial $f(s)$ with no imaginary axis roots we will need a factorisation $f(s) = f_s(s) f_a(s)$ where $f_s(s)$ is monic and has only left half plane (LHP) roots and $f_a(s)$ is monic and has only right half plane (RHP) roots. The approach used will find intervals for the coefficients of $f_s(s)$ and $f_a(s)$ which may be as tight as desired. One technique which will be used to successively narrow the intervals employs the Krawczyk operator [18], which may be computationally more efficient than one which repetitively uses a guaranteed root localisation method. A detailed algorithm may be found in [15].

7 Computation of the $\mathcal{L}_2$-norm with Guaranteed Accuracy

We begin with a problem that is rather trivial to solve in a guaranteed accuracy way, namely the $\mathcal{H}_2$-norm. Let us recall a standard state-space method for its computation. Given a stable system $G(s) = \left[\begin{array}{c|c} A & B \\ \hline C & 0 \end{array}\right]$, its $\mathcal{H}_2$-norm can be calculated by

$$\|G(s)\|_2 \quad = \quad \sqrt{\mathrm{trace}\,(CPC^*)}$$

where P is the solution of the following Lyapunov equation

$$AP + PA^* + BB^* \quad = \quad 0\,.$$

In standard approaches using floating point arithmetic rounding error is inevitable, no matter how numerically stable the routines are. This problem can in fact be avoided since the Lyapunov equation is a set of linear equations. If rational number representations are employed, computer algebra can then be used to calculate the exact solution using algebraic operations only.

When a function in $\mathcal{RL}_2$ is given exactly (with rational constants), its $\mathcal{L}_2$-norm can be computed with guaranteed accuracy. Consider a (not necessarily stable) SISO system $G(s) = \frac{n(s)}{d(s)}$ where $d(s)$ has no imaginary axis roots and the degree of $n(s)$ is strictly smaller than $d(s)$. Write $d(s) = d_s(s)d_a(s)$ where $d_s(s)$ has only LHP roots and $d_a(s)$ has only RHP roots, i.e., the stable/anti-stable factors. Then, since

$$\begin{aligned}\left(\frac{n(j\omega)}{d(j\omega)}\right)^* \frac{n(j\omega)}{d(j\omega)} \quad &= \quad \frac{n(-j\omega)}{d(-j\omega)}\frac{n(j\omega)}{d(j\omega)} \\ &= \quad \frac{n(-j\omega)}{d_s(-j\omega)d_a(j\omega)}\frac{n(j\omega)}{d_s(j\omega)d_a(-j\omega)}\,,\end{aligned}$$

the $\mathcal{L}_2$-norm of G is equal to the $\mathcal{H}_2$-norm of $\frac{n(s)}{d_s(s)d_a(-s)}$. Notice that $d_s(s)d_a(-s)$ has only LHP roots. The method of the previous paragraph along with interval matrix inversion [10] can be applied and a bound for $\|G(s)\|_2$ can be obtained with 'guaranteed accuracy' in the sense of Definition 1. This method can immediately extended to the MIMO case since, when $G(s) = (G_{ij}(s))$, $\|G(s)\|_2^2 = \sum_{i,j} \|G_{ij}(s)\|_2^2$.

8 Computation of the $\mathcal{L}_\infty$-norm with Guaranteed Accuracy

The standard approach to computing the $\mathcal{L}_\infty$-norm finds H_γ, the Hamiltonian A-matrix of $\Phi_\gamma(s) = \gamma^2 I - G^T(-s)G(s)$, and uses floating point methods to find the smallest γ for which H_γ has no eigenvalues on the imaginary axis [3, 4]. It is well known that this last step is prone to numerical difficulties. Furthermore, because of repetitive computation of eigenvalues and the largest singular values, the method does not seem suitable for implementation in a computer algebra system. Unlike the $\mathcal{L}_2$-norm computation, alternative approaches need to be taken to construct an algorithm with guaranteed accuracy.

The following approach, which involves reducing the problem to real root localisation of a polynomial rather than an eigenvalue test, does allow a guaranteed accuracy algorithm to be implemented. The idea is summarised in the following theorem.

Theorem 2 ([15]). *Let $G(s) \in \mathcal{RL}_\infty$ and assume that its $\mathcal{L}_\infty$-norm $\gamma_\infty = \|G(s)\|_\infty$ is not achieved at $s = 0$ or $j\infty$ ($\omega = 0$ or ∞), i.e., $\gamma_\infty > \overline{\sigma}(G(0))$ and $\gamma_\infty > \overline{\sigma}(G(j\infty))$ where $\overline{\sigma}(\cdot)$ is the largest singular value. Furthermore, let $\Phi_\gamma(s) = \gamma^2 I - G^T(-s)G(s)$ and denote $g_\gamma(s^2) = \det \Phi_\gamma(s)$. Moreover, write $g_\gamma(x) = \frac{n_\gamma(x)}{d_\gamma(x)}$.*

Let $h_\gamma(x)$ be the square-free part of $n_\gamma(x)$ considered as a polynomial in x and γ. Then, if $\gamma > \gamma_\infty$, $h_\gamma(x)$ has no root in $-\infty < x < 0$. Further, $h_{\gamma_\infty}(x)$ has a multiple root in $-\infty < x < 0$.

The candidate values for the $\mathcal{L}_\infty$-norm are then $\overline{\sigma}(G(0))$, $\overline{\sigma}(G(j\infty))$ and the real roots γ of the discriminant of $h_\gamma(x)$, which is a polynomial in γ^2. Note that all the candidate values can be found to desired accuracy via real root localisation methods mentioned in Section 6. The true one can be chosen from the candidates using the Sturm test. Namely, we find the candidate γ for which the corresponding upper bound (resp. lower bound) gives no (resp. some) roots of $h_\gamma(x)$ in $-\infty < x < 0$. In this way, using a computer algebra system, the $\mathcal{L}_\infty$-norm can be found with guaranteed accuracy in the sense of Definition 1.

A Maple program implementing the above method found the $\mathcal{H}_\infty$-norm of (1) to be between 10 and 10.00001 in less than 0.2 seconds on a 750 MHz Pentium. The bound indeed contains the actual value. We illustrate the method on a further specific example. The $\mathcal{L}_\infty$-norm of the plant in Example 4.2 in [32]

$$G(s) = \begin{bmatrix} \frac{s^2+0.15s+2.5}{s^4+0.35s^3+3.51s^2+0.45s+2.0} & \frac{0.2s+1.0}{2(s^4+0.35s^3+3.51s^2+0.45s+2.0)} \\ \frac{0.1s+0.5}{s^4+0.35s^3+3.51s^2+0.45s+2.0} & \frac{s^2+0.2s+1.0}{2(s^4+0.35s^3+3.51s^2+0.45s+2.0)} \end{bmatrix}$$

is found to be one of the roots of the following 12th order polynomial in γ (or 6th order in γ^2) with *integer* coefficients:

$$\begin{aligned} &1540583450598938837 3\gamma^{12} - 2070088084346678781094\gamma^{10} \\ &+ 5707237953777309755325\gamma^8 - 4082948339683566097500\gamma^6 \\ &+ 8902009499296500000000\gamma^4 - 26280511750000000000\gamma^2 \\ &+ 3240000000000000000 \ . \end{aligned}$$

Via a real root localisation method the actual $\mathcal{L}_\infty$-norm can be found with guaranteed accuracy. Specifying, say, sixteen decimal places to be accurate in advance, the $\mathcal{L}_\infty$-norm was computed to be 11.4703965432689763. *We guarantee that this answer is accurate to 16 places (which was obtained in less than 1.4 seconds)!*

9 $\mathcal{H}_2$-optimal Controller Synthesis with Guaranteed Accuracy

This section is devoted to the guaranteed solution to the normalised $\mathcal{H}_2$-optimal controller synthesis problem described in [14] in the SISO case. We discuss how to obtain bounds for the coefficients of the optimal controller with guaranteed accuracy. In the feedback configuration in Figure 1, let P be the plant to be controlled which is expressed by a strictly proper rational transfer function and let the transfer function from $w = (d_1\ d_2)^T$ to $z = (y_1\ y_2)^T$ be denoted by $T_{wz}(s)$. We consider the problem of minimising $\|T_{wz}(s)\|_2$ over all stabilising controllers K. Write $P = P_N/P_D$ where P_N and P_D are coprime polynomials. Let M_D be a stable polynomial with positive leading coefficient that satisfies $M_D^\sim M_D = P_N^\sim P_N + P_D^\sim P_D$ where $f^\sim(s) = f(-s)$ for a real polynomial $f(s)$. The coefficients of M_D can be found with guaranteed accuracy as mentioned in Section 6. Then, $P = N/M$, where $M = P_D/M_D$ and $N = P_N/M_D$, is a normalised coprime factorisation. Any stabilising rational controller can be factorised as $K = U/V$ such that $U, V \in \mathcal{RH}_\infty$ satisfy the Bezout identity $MV - NU = 1$. It is not difficult to deduce from [20, 25] that the denominators of U and V for the optimal controller are M_D. Write $U = U_N/M_D$ and $V = V_N/M_D$. U_N and V_N can be found from the Bezout identity which is now equivalent to

$$P_D V_N - P_N U_N \quad = \quad M_D^2 \ . \tag{2}$$

When the order of the plant P is n, the degrees of P_D, M_D and V_N are all n and those of P_N and U_N are at most $n-1$. It can be shown that a pair of U_N and V_N that satisfy the degree requirements and also (2) is unique and, moreover, U_N and V_N thus obtained yield the optimal controller. Equating the coefficients of (2), a matrix equation $Sb_p = b_m$ is obtained where S is a matrix whose elements consist of the coefficients of P_N and P_D, b_p is a column vector that is composed of the coefficients of U_N and V_N and b_m is a column vector whose elements are polynomials in the coefficients of M_D. S is nonsingular and hence the elements of b_p, or the coefficients of U_N and V_N, are expressed as polynomials in the coefficients of M_D. Since the coefficients of M_D can be found with guaranteed accuracy, the coefficients of U_N and V_N can also be found with guaranteed accuracy.

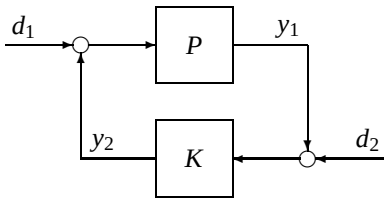

Figure 1: $\mathcal{H}_2$ -optimal Feedback Configuration

10 Synthesis Problems with Guaranteed Accuracy

The extension of Definition 1 to problems in which a vector or matrix is sought is straightforward. In the previous section bounds for the coefficients of the optimal controller were used to guarantee accuracy. However in a controller synthesis problem there may be more desirable ways to specify 'guaranteed accuracy' than in terms of the coefficients of the controller's transfer function or state-space realisation. Other possibilities might be to guarantee that the gap between the computed solution and the true solution is less than some given $\varepsilon > 0$, or that the achieved performance measure is guaranteed to be within some $\varepsilon > 0$ of the true performance measure.

However another difficulty arises when a discontinuity in the solution of a synthesis problem occurs. A synthesis problem in general consists of a multiple stage procedure. When the outputs from the previous stages are given as intervals rather than exact values, the following stage inevitably has to work with these intervals which is problematic in the case where a discontinuity occurs nearby. Evidently basic questions relating

to continuity of solution need to be answered along with algorithm development. The following example illustrates this.

Consider the plant $\frac{s+\varepsilon}{s^2+1}$ (see Section 2). With $\varepsilon = 0$, it is known from [9] that the gap-optimal controller is equal to -1. When $\varepsilon \neq 0$, the controller takes a different form. Using a specialised routine developed in Maple, the following formula was found for the controller [15]:

$$\frac{-\sigma s - \sigma a + \sigma c + 1}{s + a + c - \sigma}$$

where σ is the positive (resp. negative) root of

$$(1+\varepsilon^2)\sigma^2 + \left\{ \frac{\sqrt{-1+2\sqrt{1+\varepsilon^2}}\varepsilon^2}{\sqrt{1+\varepsilon^2}+1} - \varepsilon\left(-1+2\sqrt{1+\varepsilon^2}\right) \right\}\sigma$$
$$-\varepsilon^2 + \varepsilon\sqrt{-1+2\sqrt{1+\varepsilon^2}} - \sqrt{1+\varepsilon^2} \;=\; 0$$

for positive (resp. negative) ε, $a = \sqrt{-1+2\sqrt{1+\varepsilon^2}}$, $b = \sqrt{1+\varepsilon^2}$ and $c = \frac{(\sigma a - 1)b}{\sigma(b-1)+\varepsilon}$. The reason for the discontinuity in this case is that the top singular value in the underlying Nehari extension problem is repeated when $\varepsilon = 0$. Since the repeated root occurs at an intermediate stage in the algorithm, the algorithm needs to be able to cope with such a circumstance in a guaranteed accuracy context.

The above situation in which an actual discontinuity of solution occurs is likely to be the most challenging. Fortunately, many synthesis procedures have continuity properties, e.g. $\mathcal{H}_2$ synthesis [9] and also $\mathcal{H}_\infty$ maximum entropy [29] solutions, which makes them more promising for guaranteed accuracy solutions.

11 Concluding Remarks

The purpose of this paper is to raise the issue of validation in the context of numerical computations in systems and control and to show that progress can be made in some situations. The range of problems for which this goal can be achieved in a tractable way remains open.

For the problems investigated in this paper, numerical algorithms based on floating-point arithmetic are commercially available. A further target of this research is to tackle some problems for which satisfactory algorithms using ordinary floating-point arithmetic have proven difficult to develop. It is our view that the lack of reliable computational tools has prevented some theoretical developments from being used in practice.

The issue of computational speed is bound up with this topic in several ways, since guaranteed accuracy algorithms are likely to be more expensive in computer time. However increased computer power makes such methods a more practical proposition.

References

[1] E. Adams and U. Kulisch, editors. *Scientific Computing with Automatic Result Verification*, volume 189 of *Mathematics in Science and Engineering*, London, 1993. Academic Press.

[2] A. G. Akritas. *Elements of Computer Algebra with Applications*. John Wiley & Sons, New York, NY, 1989.

[3] S. Boyd and V. Balakrishnan. A regularity result for the singular values of a transfer matrix and a quadratically convergent algorithm for computing its L_∞-norm. *Systems and Control Letters*, 15(1):1–7, July 1990.

[4] N. A. Bruinsma and M. Steinbuch. A fast algorithm to compute the H_∞-norm of a transfer function matrix. *Systems and Control Letters*, 14(4):287–293, April 1990.

[5] G. E. Collins. Infallible calculation of polynomial zeros to specified precision. In J. R. Rice, editor, *Mathematical Software III*, pages 35–68, New York, NY, 1977. Academic Press.

[6] G. E. Collins, J. R. Johnson, and W. Küchlin. Parallel real root isolation using the coefficient sign variation method. In R. E. Zippel, editor, *Computer Algebra and Parallelism*, pages 71–87, Berlin, 1992. Springer-Verlag.

[7] G. E. Collins and W. Krandick. An efficient algorithm for infallible polynomial complex root isolation. In P. S. Wang, editor, *Proceedings of the International Symposium on Symbolic and Algebraic Computation, ISSAC '92*, pages 189–194, New York, NY, 1992. ACM Press.

[8] F. R. Gantmacher. *The Theory of Matrices*, volume 2. Chelsea Publishing Company, New York, NY, 1960.

[9] M. Green and M. C. Smith. Continuity properties of LQG optimal controllers. *Systems and Control Letters*, 26(1):33–39, September 1995.

[10] E. Hansen. Interval arithmetic in matrix computations, part I. *Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis*, 2(2):308–320, 1965.

[11] B. Hanzon and J. M. Maciejowski. Constructive algebra methods for the L_2-problem for stable linear systems. *Automatica*, 32(12):1645–1657, December 1996.

[12] H. Hoon and R. Liska, editors. *Journal of Symbolic Computation: Special Issue on Application of Quantifier Elimination*, volume 24, number 2. Academic Press, August 1997.

[13] L. Jaulin, M. Kieffer, O. Didrit, and É. Walter. *Applied Interval Analysis*. Springer-Verlag, London, 2001.

[14] E. A. Jonckheere and L. M. Silverman. A new set of invariants for linear systems — application to reduced order compensator design. *IEEE Transactions on Automatic Control*, AC-28(10):953–964, October 1983.

[15] M. Kanno. *Guaranteed Accuracy Computations in Systems and Control*. PhD thesis, University of Cambridge, 2003. In preparation.

[16] W. Krandick and S. Rump, editors. *Journal of Symbolic Computation: Special Issue on Validated Numerical Methods and Computer Algebra*, volume 24, number 6. Academic Press, December 1997.

[17] W. Krandick and S. Rump. Special issue on validated numerical methods and computer algebra: Foreword of the guest editors. *Journal of Symbolic Computation*, 24(6):625–626, December 1997.

[18] R. Krawczyk. Newton-Algorithmen zur Bestimmung von Nullstellen mit Fehlerschranken. *Computing*, 4(3):187–201, 1969.

[19] D. H. Lehmer. A machine method for solving polynomial equations. *Journal of the Association for Computing Machinery*, 8(2):151–162, April 1961.

[20] D. G. Meyer and G. F. Franklin. A connection between normalized coprime factorizations and linear quadratic regulator theory. *IEEE Transactions on Automatic Control*, AC-32(3):227–228, March 1987.

[21] R. E. Moore. *Interval Analysis*. Prentice-Hall, Englewood Cliffs, NJ, 1966.

[22] R. E. Moore. A test for existence of solutions to nonlinear systems. *SIAM Journal on Numerical Analysis*, 14(4):611–615, September 1977.

[23] N. Munro, editor. *Computing & Control Engineering Journal: Special Issue on Symbolic Computation*, volume 8, number 2. Institution of Electrical Engineers, April 1997.

[24] N. Munro, editor. *Symbolic Methods in Control System Analysis and Design*, volume 56 of *IEE Control Engineering Series*. The Institution of Electrical Engineers, Stevenage, 1999.

[25] D. Mustafa and K. Glover. *Minimum Entropy H_∞ Control*, volume 146 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, Berlin Heidelberg, 1990.

[26] A. Neumaier. *Interval Methods for Systems of Equations*. Cambridge University Press, Cambridge, 1990.

[27] D. Nešić and I. M. Y. Mareels. Controllability of structured polynomial systems. *IEEE Transactions on Automatic Control*, 44(4):761–764, April 1999.

[28] J. R. Pinkert. An exact method for finding the roots of a complex polynomial. *ACM Transactions on Mathematical Software*, 2(4):351–363, December 1976.

[29] L. Y. Wang and G. Zames. Lipschitz continuity of H^∞ interpolation. *Systems and Control Letters*, 14(5):381–387, June 1990.

[30] H. S. Wilf. A global bisection algorithm for computing the zeros of polynomials in the complex plane. *Journal of the Association for Computing Machinery*, 25(3):415–420, July 1978.

[31] M. Zettler and J. Garloff. Robustness analysis of polynomials with polynomial parameter dependency using Bernstein expansion. *IEEE Transactions on Automatic Control*, 43(3):425–431, March 1998.

[32] K. Zhou and J. C. Doyle. *Essentials of Robust Control*. Prentice-Hall, Upper Saddle River, NJ, 1998.